API Reference

The DC/OS API is a collection of routes backed by DC/OS components that are made available through an API gateway called Admin Router.

API Gateway

Admin Router is an API gateway built on top of NGINX with the following goals:

- Present a unified control plane for the DC/OS API
- Proxy API requests to component services on master and agent nodes
- Enforce user authentication
- Serve up the DC/OS GUI

Admin Router runs on each DC/OS node in one of two configurations:

- **Admin Router Master** exposes the Master Routes.

  This configuration runs on each master node and serves as the primary API gateway for interaction with DC/OS components.

- **Admin Router Agent** exposes the Agent Routes.

  This configuration runs on each agent node and provides routes for monitoring, debugging, and administration.

  Some agent routes, like logs and metrics, are proxied through the master Admin Router to allow external access. Other routes, like component management, are for internal use only.

Route Types

Admin Router exposes several types of routes:

- **Proxy Routes** retrieve resources from another URL.
- **File Routes** retrieve static files.
- **Lua Routes** execute Lua code to generate responses.

- **Redirect Routes** redirect to another URL.
- **Rewrite Routes** translate routes into other routes.

Cluster Access

To determine the URL of your cluster, see Cluster Access.

Versioning

Sections of the DC/OS API are versioned by component, route, or resource.

For details on the versioning mechanisms, see Versioning.

Authentication

Some routes are unauthenticated, but most require an authentication token.

For details on how to obtain and use an authentication token, see Authentication HTTP API Endpoint.

Authorization

Most authenticated routes also require authorization via permissions. Permissions in DC/OS Enterprise consist of a hierarchical resource identifier and an action (create, read, update, delete, full).

Permission enforcement can be performed at two levels.

- **Course-grained permissions** are enforced by Admin Router at the route level.
- **Fine-grained permissions** are enforced by individual backend component services.

Permissions Management can be performed by users with the Superuser permission using the Identity and Access Management API. Users with the Superuser permission also have implicit permission to access all routes.

Route Usage

- To determine the full URL of a API resource through a **proxy route**, join the cluster URL, route, and backend component resource path.

  `<cluster-url>/<route>/<resource-path>`

  For example, get the Mesos version from: `https://dcos.example.com/mesos/version`

- **File routes** have no backend component, but may serve a directory of files or a single file. So for file routes, specify the file path instead of the backend component resource path.

  `<cluster-url>/<route>/<file-path>`

  For example, get the DC/OS version of the cluster from: `https://dcos.example.com/dcos-metadata/dc`

- **Lua routes** immediately execute code in Admin Router without proxying to an external backend component. So for Lua routes, no path is required after the route.

  `<cluster-url>/<route>`

  For example, get the public IP of the master node and cluster ID from: `https://dcos.example.com/metadata`

- **Rewrite and redirect routes** may pass through one or more other URLs or routes before returning a resource. So for those routes, follow the chain of URLs and routes to find the endpoint. The resource path will depend on the final endpoint.

  Most rewrites and redirects terminate in another DC/OS API route, with the notable exception of `/login`, which uses OpenID Connect to authorize with an external identity provider and then redirects back to the DC/OS API.